Protecting the Content Integrity of Digital Imagery with Fidelity Preservation: An Improved Version

This copy represents the peer reviewed and accepted version of paper:

Marco Botta, Davide Cavagnino, Victor Pomponiu, "'Protecting the Content Integrity of Digital Imagery with Fidelity Preservation': an improved version ", ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM) (2014), vol. 10 no. 4 pp. 29:1-29:5

http://doi.acm.org/10.1145/2568224

'Protecting the Content Integrity of Digital Imagery with Fidelity Preservation': an improved version

MARCO BOTTA AND DAVIDE CAVAGNINO, Università di Torino
VICTOR POMPONIU, University of Pittsburgh

Fragile watermarking has attracted a lot of attention in the last decade. An interesting approach has been presented in 2011 by Lin et al. which results in very high quality of the watermarked images. However, after a thorough examination of the paper, a few improvements are proposed in our revised version of the algorithm in order to overcome some shortcomings. In particular, changes to the pseudo-code and modifications to deal with pixel saturation are suggested, along with a way to improve the scheme security. Finally, a deeper analysis of the security is presented.

Categories and Subject Descriptors: **H.4.3 [Information Systems Applications] [Information Interfaces and Presentation]**: Communications Applications; **I.4.9 [Image Processing and Computer Vision]** : Applications

General Terms: Security, Content Integrity

Additional Key Words and Phrases: Information hiding, fragile watermarking

ACM Reference Format:

1. INTRODUCTION

Digital watermarking is a set of techniques aimed at inserting a signal, called watermark, into a digital object. In particular, fragile watermarks have the objective to be modified by a minimal alteration made to the host object. In this context, they may be used for content integrity and authentication. A very interesting algorithm for the fragile watermarking of images has been recently presented in [Lin et al. 2011], that results in high quality images. Nonetheless, in our analysis of that algorithm, we found some shortcomings that need to be solved in order to use their efficient technique. In this paper, we consider these issues and suggest how they can be dealt with by proposing an improved version of Lin et al.'s algorithm.

2. THE LIN ET AL. WATERMARKING SCHEME IMPROVEMENTS

Lin et al. [2011] presented an algorithm (referred to in the following as LLC) for the detection of image alterations; the algorithm inserts a fragile watermark with the intent that a modification of the image will change some parts of the watermark allowing for the localization of the tampering. The claimed advantage of the method is that only one pixel per block needs to be eventually modified by ±1 gray levels, thus producing high quality images.

Authors' addresses: M. Botta, Dipartimento di Informatica, C.so Svizzera 185, 10149 Torino, Italy; email: marco.botta@unito.it; D. Cavagnino, Dipartimento di Informatica, C.so Svizzera 185, 10149 Torino, Italy; email: davide.cavagnino@unito.it; V. Pomponiu, Department of Radiology, 3362 Fifth Avenue, Pittsburgh, 15213, PA, USA; email: vpomponiu@acm.org.

DOI: …

One problem of LLC lies in the pseudo-code: indeed, the while loop in line 9 never terminates when f is different from the intended watermark $(l_1 , l_2 , \ldots , l_{n+1})_2$, as the value of f is not recomputed inside the loop, so the test condition never changes its truth value.

A more subtle problem of LLC is related to the modifications of the pixels. What happens when the pixel to be modified already has a value at the extreme of the allowed interval? For instance, let us suppose that the pixel to modify by –1 has value 0. The result (–1) is clearly out of the allowed range for a pixel. How the LLC algorithm deals with this case should be made explicit in order for the algorithm to be applicable to a wider range of images. There are two possible alternatives.

The first one is to use modulo arithmetic when changing the pixel value: so $p_{|d|} \pm 1$ should actually be $(p_{|d|} \pm 1)$ mod (MAX+1), where MAX is the maximum allowed value for a pixel. If this is the case, then the claim that the proposed method changes at most one pixel by ±1 per block should be revised. Moreover, the quality of the watermarked image can be affected in an unacceptable way: if the pixel value changes from 0 to 255 due to modulo arithmetic, this results in a visibly distorted pixel (salt & pepper noise). But even though the change were not visible, the PSNR of the resulting image would be about half of that expected. To make it simple, let us choose a value n so that we have only 1 block (e.g., for a 512×512 pixel image, n=18). According to Lin et al.'s claim, only one pixel at most would be changed by ±1, resulting in a PSNR > 102 dB. But if the pixel is changed by 255, then the resulting PSNR is only 54.18 dB, almost half of that theoretically expected. This is never the case for the test image (Lena) used by Lin et al., so this problem was not revealed.

ALGORITHM RLLC: The revised fragile watermarking procedure

Input: the protected image O, authentication data, parameter n, and a user key UK
Output: the watermarked image O'

1. divide O into non-overlapping blocks each containing 2^n pixels
2. set all blocks as unprocessed
3. select an unprocessed block, called B, and set B as a processed block
3.5 permute the pixels of B according to B's position in O and to UK, and produce B^p
4-6. generate watermark bits $L = [l_1 , l_2 , \ldots , l_{n+1}]$ according to B's position in O and to UK
7. let $p_1 , p_2 , \ldots ,$ and p_{2^n} be the pixels of the block B^p
7.5 repeat
8. calculate the weighted-sum function $f = \left(\sum_{m=1}^{2^n} m \times p_m\right) \bmod 2^{n+1}$
9. if $f \neq (l_1 , l_2 , \ldots , l_{n+1})_2$
$d = (l_1 , l_2 , \ldots , l_{n+1})_2 - f$
if $(0 < |d| \leq 2^n)$
$$\begin{cases} \boldsymbol{p_{|d|} + 1}, & \textbf{if } (d > 0 \textbf{ and } p_{|d|} < \text{MAX}) \textbf{ or } (d < 0 \textbf{ and } p_{|d|} = 0) \\ \boldsymbol{p_{|d|} - 1}, & \textbf{if } (d < 0 \textbf{ and } p_{|d|} > 0) \textbf{ or } (d > 0 \textbf{ and } p_{|d|} = \text{MAX}) \end{cases}$$
else
$$\begin{cases} \boldsymbol{p_{2^{n+1}-|d|} - 1}, & \textbf{if } (d > 0 \textbf{ and } p_{2^{n+1}-|d|} > 0) \textbf{ or } (d < 0 \textbf{ and } p_{2^{n+1}-|d|} = \text{MAX}) \\ \boldsymbol{p_{2^{n+1}-|d|} + 1}, & \textbf{if } (d < 0 \textbf{ and } p_{2^{n+1}-|d|} < \text{MAX}) \textbf{ or } (d > 0 \textbf{ and } p_{2^{n+1}-|d|} = 0) \end{cases}$$
end-if
end-if
9.5 until $f = (l_1 , l_2 , \ldots , l_{n+1})_2$
9.6 restore the pixels of B^p into their original positions to produce B' and save it in O'
10. check whether there exists any unprocessed block. If so, go to Step 3.
11. return the watermarked image O'

The second alternative is to use *saturated* arithmetic, i.e. any value out of range is saturated to the extreme of the range. In this case, the loop in line 9 of LLC would never end, even if it were correct. A straightforward solution would be to select two pixels in positions j and k, such that $(\alpha j + \beta k) \bmod 2^{n+1} =$

$|d|$, being α and β the modifications to apply to the two selected pixels (with opposite sign if d is negative). This solution is always feasible, but it might be computationally expensive to select appropriate j and k, and, moreover, α and β might be larger than ±1.

The RLLC algorithm presents a neat solution that uses a different approach and produces high quality watermarked images, by allowing to change by ±1 two or more pixels (a similar approach is described in [Zhang and Wang 2006] but only with an example and without a formal proof of termination). The only issue here is that more than one pixel may be changed by one gray level, slightly reducing the PSNR from the values computed in [Lin et al. 2011].

Formally, we can prove that the RLLC algorithm always terminates by considering these two cases:

a) if the pixel has to be increased by 1 and its value is less than MAX, or has to be decreased by 1 and it is greater than 0, then the new pixel value will make $f = (l_1 , l_2 , \ldots , l_{n+1})_2$ as in the LLC algorithm, so the loop will terminate and the correct value will be stored in the block;

b) if the pixel is equal to 0 (or to MAX) then it is increased (or decreased, respectively) by 1, moving its value away from the range limit (i.e. away from 0 and MAX); in the next repeat-until loop, if the new selected pixel satisfies case a) then the loop will terminate, otherwise also the new pixel will be moved away from 0 or MAX, changing once more the difference d. Then the loop will be repeated, but case b) cannot happen forever, because in the worst case 2^n pixels will be moved away from the range limits, and then only case a) will apply, terminating the loop.

In order to assess the quality of the images watermarked with our proposed revision, Table I reports the average values and standard deviations computed over a database of 1000 512×512 pixels images taken from [Li et al. 2007], comparing the theoretical properties of the approach with the real values obtained by Lin et al.'s algorithm (correctly implemented and using modulo arithmetic) and by our revised version. We focused on values of n used in practical watermarking applications.

Table I. Comparison of the quality between LLC (with modulo arithmetic) and RLLC algorithms. (mean ± std).

	Theoretical Values		LLC (modulo arithmetic)				RLLC	
n	Max Modified pixels (by ±1)	Theoretical Min PSNR	PSNR	Actually Modified Pixels	Average modification	# pixels modified by 255	PSNR	Actually Modified Pixels
5	8192	63.18	51.93±11.83	8064.72±14.49	2.27±4.08	40.3±129.96	63.23±0.07	8107.18±142.09
6	4096	66.19	56.28±11.24	4064.24±6.54	1.77±3.3	12.4±52.8	66.21±0.06	4078.11±59.13
7	2048	69.2	59.55±11.55	2040.21±2.92	1.81±3.3	6.54±26.51	69.2±0.06	2047.62±29.83
8	1024	72.21	63.4±12	1023.65±6.46	1.86±3.03	3.56±12.79	72.21±0.06	1026±15.03
9	512	75.22	67.7±12.1	512.78±3.93	1.83±2.52	1.71±5.35	75.21±0.06	513.49±7.05

RLLC effectively modifies pixels by at most ±1, while LLC modifies a large number of pixels by 255. On average, RLLC modifies a slightly larger number of pixels than LLC, but its average PSNR is greater than the theoretical minimum value, as expected, and it is very sharp (std is 0.06), while LLC is far from the theoretical value, even though for larger values of n this difference decreases.

3. DISCUSSION AND CONCLUSIONS

First of all, let us examine the security of Lin et al.'s algorithm: in their paper, they declare that "without the user key *UK* and the parameter n, the intruder is incapable of successfully manipulating the watermarked image and passing the verification procedure".

In our opinion, the method simply remaps bits to bits, without increasing the fragility of the watermark, which instead depends on the length of L: an attacker will not try to break either A or *UK*, but only to discover the value of L. In this case, only L is useful for the security of the method. Indeed,

an intruder does not need to know the user key, nor parameter n, as claimed by Lin et al. to successfully tamper the watermarked image (it is secure only if n=0).

Here is an extremely simple procedure to tamper an image watermarked with LLC and pass the verification test. Firstly, let us consider the simpler case where the intruder actually knows parameter n. (S)He can therefore split the image in blocks of size 2^n pixels as done by LLC. Then, $f' = \left(\sum_{m=1}^{2^n} m \times p_m\right) \bmod 2^{n+1}$ can be computed for each block and from that the lock matrix L' is derived. Now, the intruder modifies pixel p_i by +2 and pixel p_{2i} by −1, $1 \le i \le 2^{n-1}$. As can be easily verified, the value of f' does not change and the watermarked image so tampered will pass the verification procedure. Then, let us consider the case in which the intruder does not know parameter n, but (s)he supposes that n>0. The intruder cannot split the watermarked image into blocks as done by the LLC algorithm, but this is not necessary to tamper the image: whatever the value of parameter n, the LLC algorithm will put into the first block the first 2 pixels of the host image that will be used in the computation of function f for the first block. By changing these 2 pixels in the watermarked image the way we described above, the value of f' will not change, so the extracted lock matrix L' will be the same as the one inserted. Again, this tampering will go undetected by the verification process.

Finally, Lin et al. claim that the probability to successfully counterfeit a watermarked image (of H×W pixels) without the knowledge of UK is $1/[(n+1)!]^{((H\times W)/2^n)}$ when n≥1. Actually, this probability is 1, as the tampering procedure shown always counterfeits the watermarked image whenever the value of n is known. Moreover, since n is limited by the image size ($n \le \log_2(H \times W)$), there is a probability of $1/n$ to guess the value of n, and successfully apply the tampering procedure to every block.

One way to make the scheme more secure is to reorder the pixels that will form a block according to UK (step 3 of LLC): as an intruder does not know UK, he will not be able to apply the same permutation on the watermarked image, even when knowing parameter n. The probability to successfully tamper a block is now $1/[2^n(2^n-1)]$. Step 3.5 of the RLLC algorithm permutes the pixels of every block of the host image according to the block position and to the user key UK, and step 9.6 reorders the pixels of all blocks into their original positions.

One last observation shows that LLC has a behavior typically opposite to other approaches. In fact, by increasing the number of bits s=n+1 inserted into a block also the probability of not detecting a tampered image (false negative) increases. The probability of a false negative for one single block is $1/2^s$. According to LLC, for an image containing H×W pixels, the total number of blocks is $b = (H\times W)/2^{s-1}$. Thus, the probability of a false negative for an image is $P_f = \left(\frac{1}{2^s}\right)^b = \left(\frac{1}{2^s}\right)^{\frac{H\times W}{2^{s-1}}}$. This function exponentially increases for increasing values of s, meaning that inserting more bits into a block increases the probability of not detecting a forged image. The explanation is that an increase in the number of bits-per-block (bpb) implies larger blocks: one more bpb implies doubling the size of the block, thus reducing the overall watermark size (improving the resulting image quality), and consequently increasing P_f. Moreover, a larger block size means a reduced localization capability of the forged areas, suggesting the use of small blocks.

REFERENCES

I. J. Cox, M. L. Miller, J. A. Bloom, J. Fridrich and T. Kalker. 2008. *Digital Watermarking and Steganography 2nd edition*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

L.-J. Li, G. Wang and L. Fei-Fei. 2007. OPTIMOL: automatic Object Picture collecTion via Incremental MOdel Learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1-8.

P.-Y. Lin, J.-S. Lee and C.-C. Chang. 2011. Protecting the content integrity of digital imagery with fidelity preservation. *ACM Trans. Multimedia Comp. Commun. and Appl.* 7, 3, Article 15 (August 2011), 15:1-15:20.

M. Wu and T. Hwang. 1984. Access Control With Single-Key-Lock. *IEEE Trans. on Softw. Eng.* 10, 2, 185-191.

X. Zhang and S. Wang. 2006. Efficient Steganographic Embedding by Exploiting Modification Direction. *IEEE Communications Letters* 10, 11, 781-783.

Received …; revised …; accepted …